IN THE CLAIMS

A marked up version of the claims as amended is set forth below.

Please amend claims 1-61 to the following:

1. (Original) A method comprising:

   receiving a first system management interrupt (SMI);

   handling the first SMI with a first processor;

   generating a wake-up signal with the first processor;

   awakening a second processor, based on the wake-up signal from the first processor; and

   handling the first SMI with a second processor.

2. (Original) The method of claim 1, wherein the first and second processors are logical processors.

3. (Original) The method of claim 1, wherein the first and second processors are physical processors.

4. (Original) The method of claim 1, wherein handling the first SMI with a first processor comprises: executing a default SMI handler located at a first memory address.

5. (Original) The method of claim 4, wherein the wake-up signal comprises a vector based on the first memory address.

6. (Original) The method of claim 5, wherein the first memory address is aligned.

7. (Original) The method of claim 6, wherein the first memory address is 4k aligned.

8. (Original) The method of claim 5, wherein handling the first SMI with the second processor comprises executing the default SMI handler located at the first memory address.

9. (Original) The method of claim 8, wherein handling the first SMI with the second processor further comprises patching an instruction pointer to a second memory address.

10. (Original) The method of claim 9, wherein the second memory address is a non-aligned address.

11. (Original) The method of claim 10, further comprising: executing code at the second memory address after the second processor has handled the first SMI.

12. (Original) A method comprising:

receiving a first system management interrupt (SMI);

executing code at a first memory address with a first processor in response to the first SMI;

generating a wake-up signal with the first processor;

awakening a second processor, based on a wake-up signal from the first processor; and

executing the code at the first memory address with the second processor.

13. (Original) The method of claim 12, wherein the wake-up signal is based on the first memory address.

14. (Original) The method of claim 13, wherein the first memory address is the location of default SMI handling code.

15. (Original) The method of claim 14, wherein the first memory address is located in conventional memory.

16. (Original) The method of claim 15, wherein the first memory address is aligned.

17. (Original) The method of claim 12, wherein both the first and second processors are logical processors located on the same die.

18. (Original) The method of claim 12, wherein the first and second processors are physical processors located on separate packages.

19. (Original) The method of claim 12, further comprising

patching an instruction pointer for the second processor to a second memory address.

20. (Original) The method of claim 19, wherein the second memory address is a non-aligned address.

21. (Original) The method of claim 20, further comprising:

executing code at the second memory address after patching the instruction pointer to the second memory address.

22. (Original) The method of claim 12, further comprising generating the SMI before receiving the SMI with a first and a second processor.

23. (Original) The method of claim 22, wherein generating the SMI comprises changing the logic level of a pin coupled to a controller hub.

24. (Original) The method of claim 22, wherein an APIC is used to generate the SMI.

25. (Original) The method of claim 24, wherein the APIC is located in the first processor.

26. (Currently Amended) A method comprising:

receiving ~~an~~ a system management interrupt (SMI);

executing a SMI handler to handle a SMI for a first processor; and

executing the SMI handler to handle the SMI for a second processor.

27. (Original) The method of claim 26, wherein the SMI is a software generated SMI.

28. (Original) The method of claim 27, wherein the first processor executes the SMI handler to handle the SMI for the first and the second processor.

29. (Original) The method of claim 26, wherein the SMI handler is located at a first memory address.

30. (Original) The method of claim 29, wherein the first memory address is a default offset from a first system management base (SMBase) address for the first processor.

31. (Original) The method of claim 30, wherein handling the second SMI with the first SMI handler comprises:

changing a target SMBase of the SMI handler from the first SMBase to a second SMBase for the second processor; and

executing the first SMI handler using the second processor's SMBase as the target SMBase.

32. (Original) A method comprising:

executing system management interrupt (SMI) code with a first processor to handle a SMI for the first processor;

checking if the SMI is a software generated SMI; and

executing the SMI code to handle the SMI for a second processor, if the SMI is software generated.

33. (Original) The method of claim 32, wherein the first processor executes the SMI code to handle the SMI for the second processor, if the SMI is software generated.

34. (Original) The method of claim 32, wherein the second processor executes the SMI code to handle the SMI for the second processor, if the SMI is software generated.

35. (Original) The method of claim 32, wherein the first processor has a first system management base (SMBase) address.

36. (Original) The method of claim 33, wherein the second processor has a second SMBase address.

37. (Original) The method of claim 36, wherein said SMI code is located at first memory location, which has an offset from the first SMBase address.

38. (Original) The method of claim 37, wherein executing said SMI code to handle the SMI for the second processor comprises:

changing the target SMBase of the SMI handler from the first SMBase to the second SMBase; and

executing the SMI code using the second processor's SMBase as the target SMBase.

39. (Original) The method of claim 38, further comprising returning the target SMBase of the SMI handler to the first SMBase after executing the SMI code to handle the SMI for the second processor.

40. (Currently Amended) An apparatus comprising:

a controller to generate a first system management interrupt (SMI);

a first logical processor, coupled to the controller, ~~to generate a first SMI~~ to handle the first SMI and generate a wake-up signal; and

a second logical processor, coupled to the controller, to handle the first SMI after the wake-up signal is received from the first logical processor.

41. (Currently Amended) The apparatus of claim 40, wherein ~~to handle~~ handling the first SMI with the first logical processor, the first logical processor executes code at a first memory location.

42. (Original) The apparatus of claim 41, wherein the first memory address is 1k aligned.

43. (Original) The apparatus of claim 41, wherein handling the first SMI with the second logical processor comprises executing code at the first memory location.

44. (Original) The apparatus of claim 43, wherein handling the first SMI with the second logical processor further comprises patching an instruction pointer to a second memory address.

45. (Original) A system comprising:

a controller hub to generate a first system management interrupt (SMI);

a memory with a first memory address that contains code;

a first processor coupled to the controller hub to handle the first SMI, wherein the first processor executes the code at the first memory address and generates a wake-up signal; and

a second processor coupled to the controller hub to handle the first SMI after receiving the wake-up signal, wherein the second processor executes the code at the first memory address.

46. (Original) The system of claim 45, wherein the first and second processors are logical processors on the same die.

47. (Original) The system of claim 45, wherein the first and second processors are physical processors located on separate packages.

48. (Original) The system of claim 45, wherein a pin is toggled on the controller hub to generated the first SMI.

49. (Original) The system of claim 45, wherein code is executed by the controller hub to generate the first SMI.

50. (Original) The system of claim 45, wherein the code at the first memory address is SMI handling code.

51. (Original) The system of claim 50, wherein the wake-up signal is a vector containing the first memory address.

52. (Original) The system of claim 51, wherein handling the first SMI with the second processor after receiving the wake-up signal further comprises setting a pointer to a second memory address.

53. (Original) The system of claim 52, wherein the second processor, upon resuming from handling the SMI, executes code at the second memory address.

54. (Currently Amended) ~~An~~ A system comprising:

a memory with a first memory address having system management interrupt (SMI) code;

a first processor to execute the SMI code when a SMI is received; and

a second processor to execute the SMI code, if the SMI is software generated.

55. (Original) The system of claim 54, wherein the first processor has a first system management base (SMBase) address.

56. (Original) The system of claim 55, wherein the second processor has a second SMBase address.

57. (Original) The system of claim 56, wherein the first memory address has an offset from the first SMBase.

58. (Original) The system of claim 57, wherein the target SMBase by default is the first SMBase.

59. (Original) The system of claim 58, wherein the target SMBase is changed to the second SMBase before the second processor executes the SMI code.

60. (Original) The system of claim 54, wherein the first and second processors are logical processors.

61. (Original) The system of claim 54, wherein the first and second processors are physical processors.